**Q1) In your own words, describe what hashing is in general.**

Hashing is a process to convert a given string of data or a key into another value. It is used to encrypt/verify data. Hashing creates new values using different hashing algorithms. A good hash uses one-way hashing algorithms to prevent converting the hash back to the original data/key.

**Q2) Research hashing algorithms. Describe advantages and disadvantages for at least three different hashing algorithms. Please provide references for external resources.**

Three well-known hashing algorithms are MD-5, SHA256, and Whirlpool.

MD-5 stands for Message-Digest Algorithms 5. It is a one-way cryptographic function that converts any messages into an output of a fixed length of 32 characters. Compared to other hashing algorithms, the output is 32 characters, which makes it easier to compare and store the hashes. It is also useful when comparing files or codes to identify any changes. However, MD-5 is considered relatively slow compared to other algorithms. It is much less secure and vulnerable to collisions. And it is easy to obtain the same hash function (or output) for two different inputs. Hash is supposed to be a unique value, but this hashing algorithm can create the same hash for different inputs. Therefore, it is not secure or very reliable.

SHA256 is a type of SHA-2, which stands for Secure Hash Algorithm 2. SHA-2 is a newer and more secure version of SHA-1 but has become obsolete and vulnerable to attacks. While the MD-5 hash is 32 characters, SHA256 produces a longer hash (256 bits equals 32 bytes, which equals 64 hexadecimal characters). Therefore, SHA256 is more resistant to collisions and brute-force attacks. There are no known vulnerabilities or weaknesses with SHA256.

Whirlpool is an iterated cryptographic hash function. It was derived from square and Advanced Encryption Standard. It is a block cipher hash function designed after a square block cipher. It takes less than 2^256 bits length input and converts it into a 512-bit hash. (512 bit -> 64 bytes -> 128 characters) Every block cipher in whirlpool is a 8 by 8 matrix. The state of the function changes in every round by using four operations.

References

1. *‘Hashing Algorithm Overview: Types, Methodologies & Usage’,* Okta,

<https://www.okta.com/identity-101/hashing-algorithms/>

1. *‘Hash Algorithm Comparison: MD5, SHA-1, SHA-2 & SHA-3’, CodeSigningStore*

<https://codesigningstore.com/hash-algorithm-comparison>

1. *‘What are the advantages and disadvantages of SHA256 over MD5 and SHA?’,* LinkedIn community

<https://www.linkedin.com/advice/1/what-advantages-disadvantages-sha256-over-md5-sha-skills-encryption>

**Q3) Provide a stepwise description (algorithmic) of a) how you can store a password safely using hashing techniques and b) how you can verify that some string is the right password?**

1. Firstly, choose a hashing technique (hashing algorithm).
2. Use a hashing algorithm to hash the input (password)
3. Store the hashed password

When a user attempts to log in, the input will go through a hashing algorithm and compare it to the stored hashed password. If the hashes match, we can verify the input(string) is the correct password.

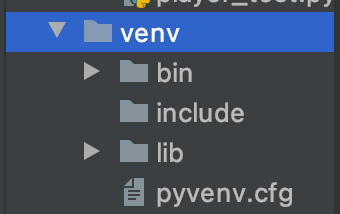
**Q4) What is the purpose of a “salt” when hashing a password? What are the two most important properties of a “salt”?**

Salt refers to adding random data to a hash function to obtain a unique output. If there are duplicates of hashed passwords in a database, the security of those hashes is fragile. So, by salting the password, which is done by either prepending or appending them two, then going through the hash function, it creates a unique output(hash). It is crucial to generate a good salt. When we add salt, it should be cryptographically solid and credential-specific. We can follow OWASP guidelines to implement credential-specific salt properly. Having a system-wise salt is pointless to mitigate attacks. It just makes the password long.

**Q5a) Install the external package argon2-cffi from pypi using pip or your IDE. Document how you installed this package.**

First, it is a good practice to create a virtual environment.

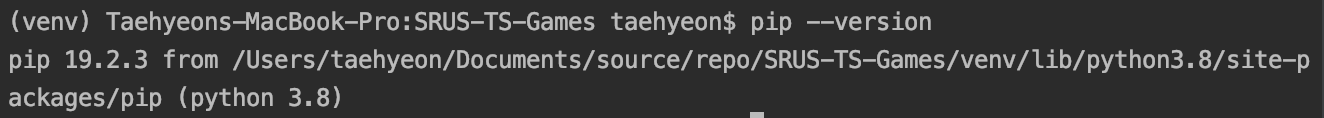




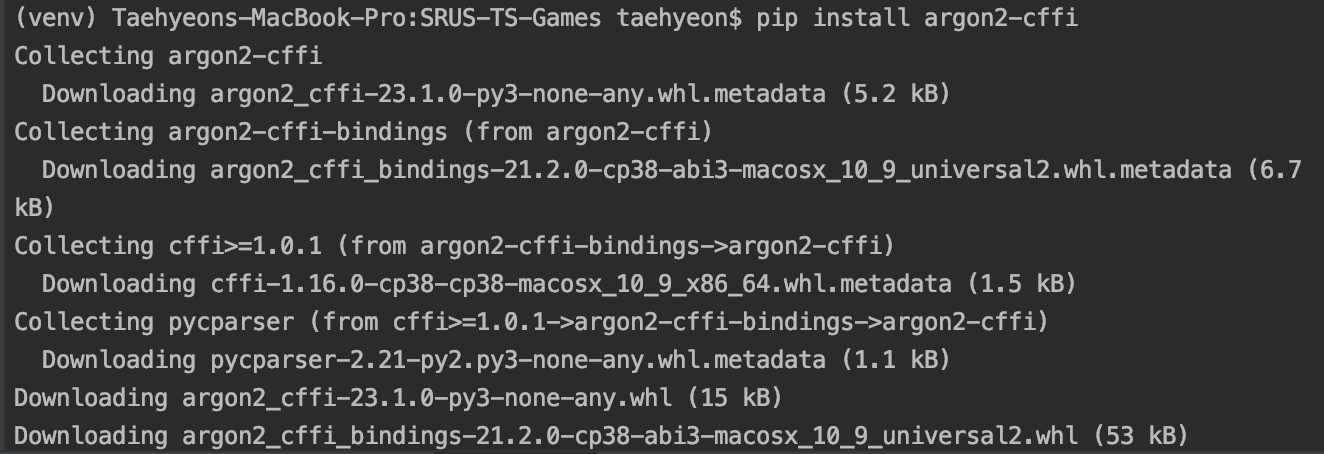
Then we activate the venv.



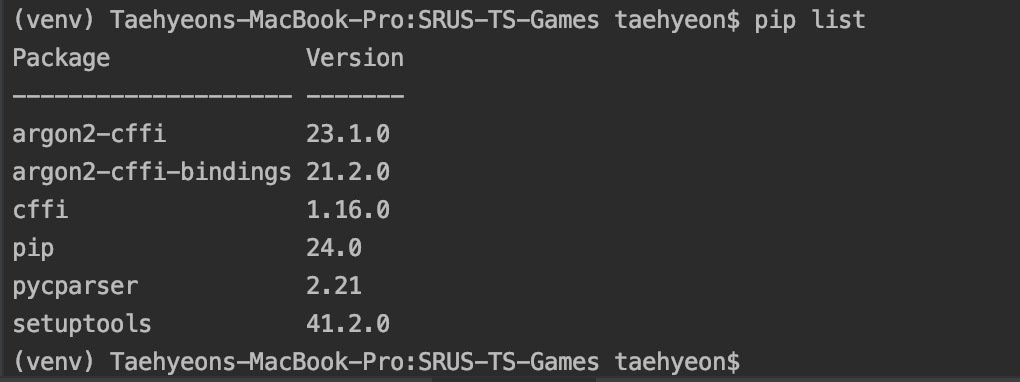
We check if the pip is installed.



We use the pip install command to install the following package.



Then, use the pip list command to check if it is installed.



**5b) Read the documentation for this package to understand what this package has to offer.**

Argon2-cffi package allows the programmer to use Argon2 hashing algorithm. It is known as a state-of-the-art password hashing algorithm. Argon2 is specifically designed for password hashing and is resistant to various attacks, including brute-force and dictionary attacks.

With argon2, we can securely store hashed passwords or other sensitive data in the databases.

It is compatible with multiple platforms like macOS, Linux and Windows.

**5g) Answer the question “How does the argon2-cffi package handle salt?”**

It was very intriguing when I attempted to compare the hashed input password with the hashed saved password using an if statement and found that the hashes were not equal. The argon2-cffi package automatically adds salt before the hashing, resulting in different hash outputs for the same input. As a result, I had to use verify method from PasswordHasher class, passing the hashed value and a non-hashed string value to check for equality.